CLAIMS

What is claimed is:

1. A server cluster having a plurality of nodes, wherein each node of said plurality of nodes comprises:

a distributor component for distributing a request to a specific node of said plurality of nodes;

a dispatcher component comprising routing information for said plurality of nodes and replicated across said plurality of nodes, wherein said routing information indicates which said node of said plurality of nodes is for processing said request, said dispatcher component coupled to said distributor component; and

a server component for processing said request, said server component coupled to said dispatcher component;

wherein said plurality of nodes are coupled to a network.

2. A server cluster as recited in Claim 1 wherein said server cluster is a web server cluster.

3. A server cluster as recited in Claim 1 further comprising a set of base files, wherein said base files are a set of frequently accessed files fitting into a cluster memory of said server cluster.

4. A server cluster as recited in Claim 3 wherein said cluster memory is a combined random access memory of each of said nodes of said server cluster.

5. A server cluster as recited in Claim 1 wherein each of said plurality of nodes further comprises a set of core files and a set of partitioned files.

6. A server cluster as recited in Claim 5 wherein said set of core files comprises a set of most frequently accessed files of said set of base files.

7. A server cluster as recited in Claim 5 wherein said set of core files is identified by the steps of:

a) logically partitioning said base files into a first subset of files having a first size, a second subset of files having a second size, and a third subset of files having a third size, wherein said base files comprising each of said first subset of files, said second subset of files, and said third subset of files are ordered in decreasing frequency of access;

b) identifying said first subset of files and said second subset of files wherein the total of said second size added to the product of said number of nodes multiplied by said first is less than said cluster memory; and

c) minimizing a total overhead due to the base files wherein said total overhead equals an overhead of said first subset plus an overhead of said second subset plus said overhead of said third subset.

8. A method for managing request distribution to a set of files stored on a server, said method comprising the steps of:

a) receiving a request for a file at a first node of a plurality of nodes, each of said nodes comprising a distributor component for distributing a request to a specific node of said plurality of nodes, a dispatcher component comprising routing information for said plurality of nodes and replicated across said plurality of nodes, and a server component for processing said request;

b) provided said request is for a core file, processing said request at said first node;

c) provided said request is for a partitioned file, determining whether said request is assigned to be processed by said first node;

c1) provided said request is for a partitioned file assigned to be processed by said first node, processing said request at said first node;

c2) provided said request is for a partitioned file assigned to be processed by another node of said plurality of nodes, forwarding said request to a specific node of said plurality of nodes as indicated by said dispatcher component of said first node and processing said request at said specific node; and

d) provided said request is not for a said core file or a said partitioned file, processing said request at said first node.

9. A method as recited in Claim 8 wherein said server is a web server cluster.

10. A method as recited in Claim 8 further comprising a set of base files, wherein said base files are a set of frequently accessed files fitting into a cluster memory of said web server cluster.

11. A method as recited in Claim 10 wherein said set of base files comprises a set of core files comprising said core file, a set of partitioned files comprising said partitioned file, and a set of on disk files.

12. A method as recited in Claim 8 wherein each of said plurality of nodes further comprises a set of core files comprising said core file and a set of partitioned files comprising said partitioned file.

13. A method as recited in Claim 12 wherein said set of core files comprises a set of most frequently accessed files of said set of base files.

14. A method as recited in Claim 12 wherein said set of core files is identified by the steps of:

a) logically partitioning said base files into said set of core files having a core size, said set of partitioned files having a partitioned size, and a set of on disk files having an on disk size, wherein said base files comprising each of said set of core files, said set of partitioned files, and said set of on disk files are ordered in decreasing frequency of access;

b) identifying said set of core files and said set of partitioned files wherein the total of said partitioned size added to the product of said number of nodes multiplied by said core size is less than said cluster memory; and

c) minimizing a total overhead due to the base files wherein said total overhead equals an overhead of said core set of files plus an overhead of said partitioned set of files plus said overhead of said on disk set of files.

15. A method for identifying a set of frequently accessed files on a server cluster comprising a number of nodes, said method comprising the steps of:

a) defining a set of base files, wherein said base files are a set of frequently accessed files fitting into the cluster memory of said server cluster, said base files ordered in decreasing frequency of access;

b) logically partitioning said base files into a first subset of files having a first size, a second subset of files having a second size, and a third subset of files having a third size, wherein said base files comprising each of said first subset of files, said second subset of files, and said third subset of files are ordered in decreasing frequency of access;

c) identifying said first subset of files and said second subset of files wherein the total of said second size added to the product of said number of nodes multiplied by said first is not greater than said cluster memory; and

d) minimizing a total overhead due to the base files wherein said total overhead equals an overhead of said first subset plus an overhead of said second subset plus said overhead of said third subset.

16. A method as recited in Claim 15 wherein said server cluster is a web server cluster.

17. A method as recited in Claim 15 wherein said cluster memory is a combined random access memory of each of said nodes of said web server cluster.

18. A method as recited in Claim 15 wherein said first subset of files is a set of core files and wherein said first size is a core size.

19. A method as recited in Claim 15 wherein said second subset of files is a set of partitioned files and wherein said second size is a partitioned size.

20. A method for determining a set of $Files_{core}$, said method comprising the steps of:

a) defining a set of *BaseFiles* as a set of frequently accessed files fitting into a *ClusterRAM*, said *BaseFiles* ordered in decreasing frequency of access;

b) logically partitioning said *BaseFiles* into a $Files_{part}$, a $Files_{core}$ and a $Files_{on\ \mathrm{disk}}$ wherein $BaseFiles = Files_{part} + Files_{core} + Files_{on\ \mathrm{disk}}$;

c) identifying said set $Files_{part}$ and said set $Files_{core}$, according to $N \bullet Size_{core} + Size_{part} \leq ClusterRAM$; and

d) minimizing $OH_{BaseFiles}$ according to $OH_{BaseFiles} = OH_{part} + OH_{core} + OH_{on\ disk}$.